OCR
Oxford Cambridge and RSA
An OCR endorsed teaching and learning tool

OCR
A Level
Computer Science

H446 – Paper 1

1

Compression, encryption and hashing

Unit 4
Exchanging data

PG ONLINE

Objectives

- Understand the difference between lossless and lossy compression
- Explain run length encoding and dictionary based compression
- Define symmetric and asymmetric encryption
- Understand how and why hashing may be used to encrypt data

Data transfer and storage

- Data is constantly being moved around computer systems and networks
  - Transfer is usually high-speed and accurate
  - As distances get longer, transfer is slower and more susceptible to interference
  - Storage space can be limited

Reducing data requirements

- Text, image and sound data can be significantly reduced in size
- Reducing the amount of data to send or store ensures that:
    - Data is sent more quickly
    - Less bandwidth is used as transfer limits may apply
    - Buffering on audio and video streams is less likely to occur
    - Less storage is required

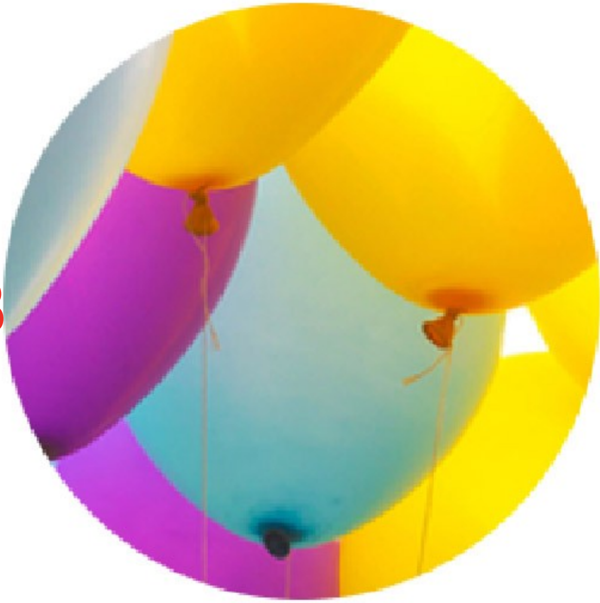
Compressing data

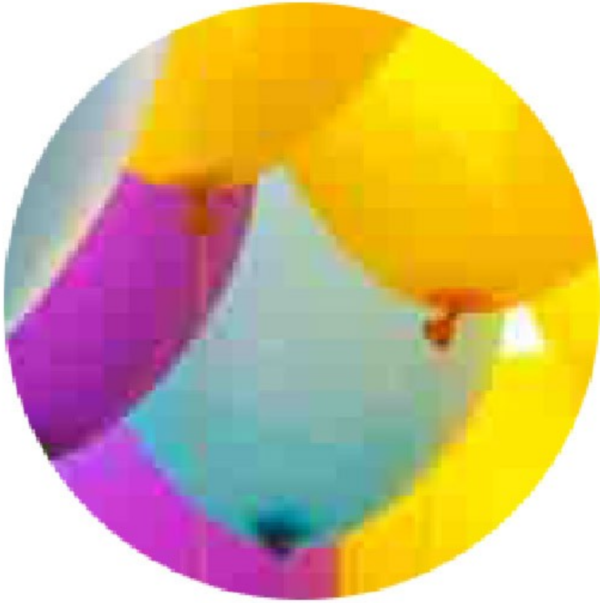
- There are two different types of compression:
  - Lossy: Non-essential data is permanently removed, for example, different shades of the same colour in an image or frequencies of sound outside the range of human hearing
  - Lossless: Patterns in the data are spotted and summarised in a shorter format without permanently removing any information

Lossy compression – JPG

- Removes data permanently to reduce file size
- Tries to reconstruct an image without the missing data
  - W l file si

120 KB

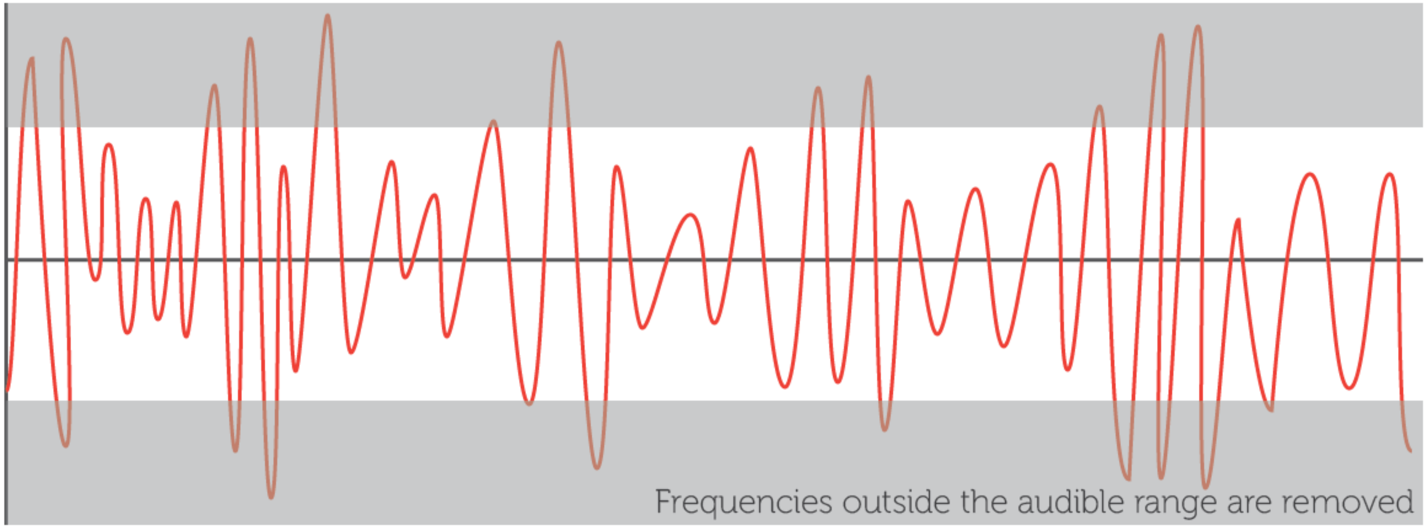




3.8 KB

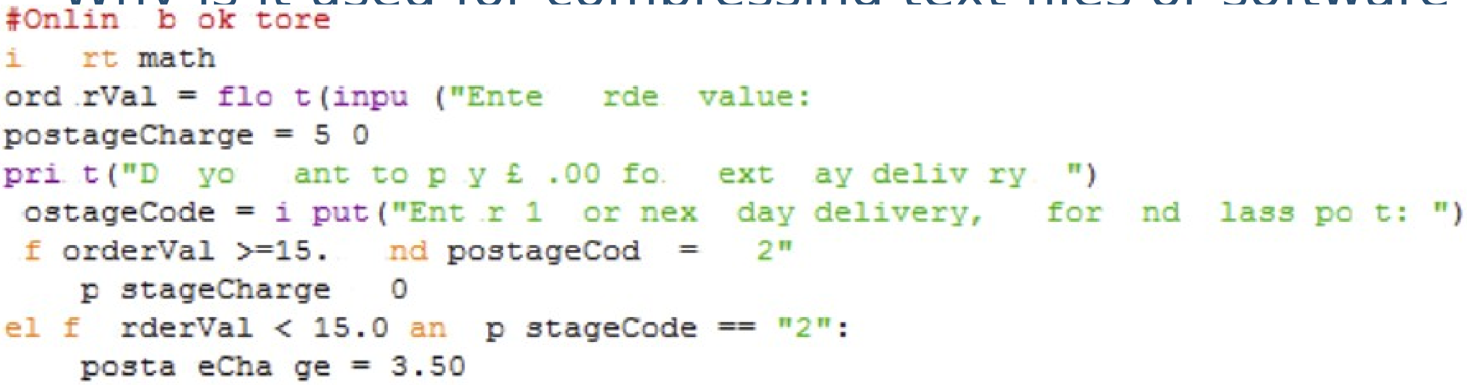
Lossy compression – MP3

- Lossy compression removes the sounds in the frequency ranges that we can’t so easily hear or that least affect the perceived playback quality
- Quieter notes played at the same time as louder sounds are also removed



Lossless compression

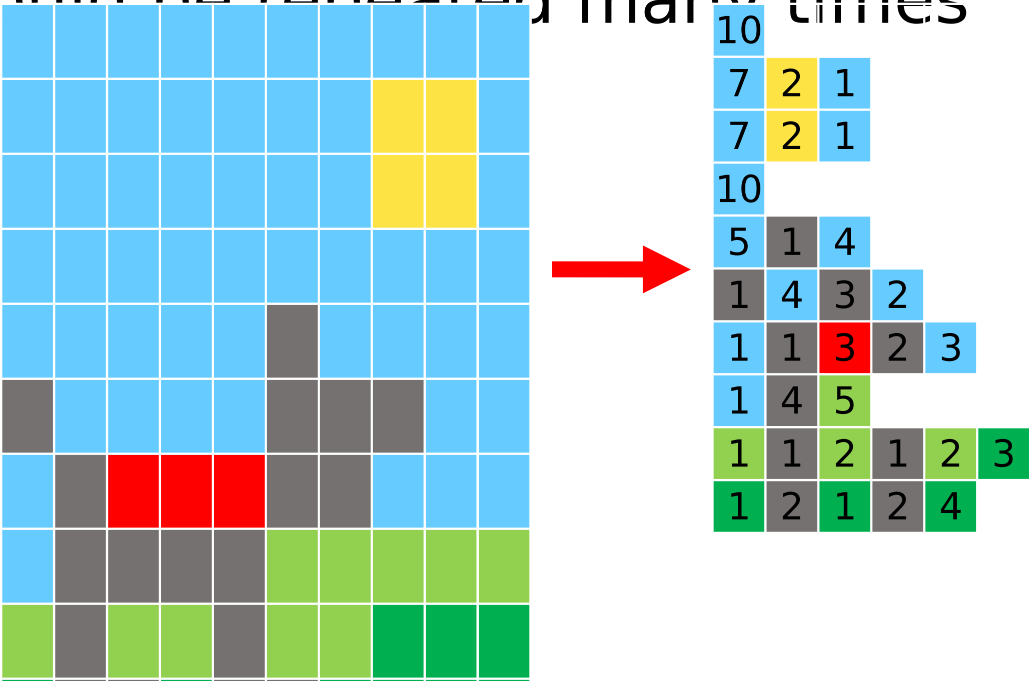
- Lossless compression works by recording patterns in the data rather than the data itself
  - Using this pattern information, a new file can be replicated exactly without any loss of data
  - The reduction in file size is less than for lossy compression
  - Why is it used for compressing text files or software



```
#Onlin  b ok tore
i   rt math
ord rVal = flo t(inpu ("Ente   rde  value:
postageCharge = 5 0
pri t("D  yo   ant to p y £ .00 fo   ext  ay deliv ry  ")
 ostageCode = i put("Ent r 1  or nex  day delivery,   for  nd  lass po t: ")
 f orderVal >=15.   nd postageCod  =   2"
    p stageCharge   0
el f  rderVal < 15.0 an  p stageCode == "2":
    posta eCha ge = 3.50
```

Run Length Encoding (RLE)

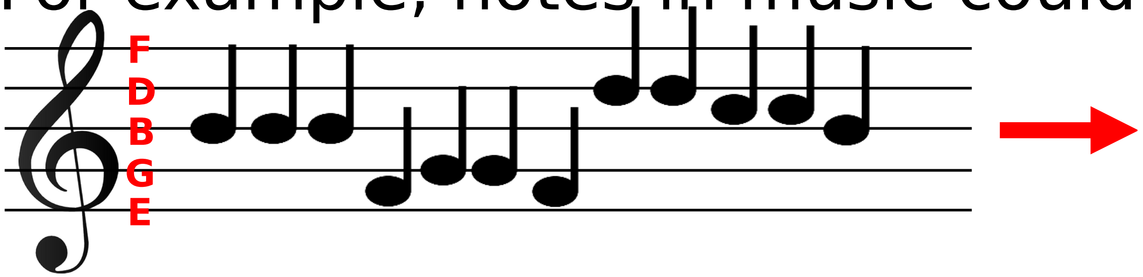
- A basic method of compression that summarises consecutive patterns of the same data
- Works well with image and sound data where data could be repeated many times



10
7 2 1
7 2 1
10
5 1 4
1 4 3 2
1 1 3 2 3
1 4 5
1 1 2 1 2 3
1 2 1 2 4

RLE of sound

- A sound recording could have many thousands of samples taken every second (typically 44,000)
  - The same sound or note played for a fraction of a second could result in hundreds of identical samples
  - RLE records one example of the sample and how many times it consecutively repeats
- For example, notes in music could be reduced:



3 B 1 F 2 G 1 F
2 D 2 C 1 B

Dictionary compression

- Spots regularly occurring data and stores it separately in a dictionary
  - The reference to the entry in the dictionary is stored in the main file thereby reducing the original data stored
  - Even though the dictionary produces additional overheads the space saving negates this problem

Forming a dictionary

- Compress the phrase “*no pain no gain* ” [15 bytes]
  - Split the pl ds or characters

Number	Entry	Binary
1	no_	00
2	p	01
3	ain_	10
4	g	11

- Using the simple dictionary example above:

“*no pain no gain* ” = 0001 1000 1110 = [12 bits]

Compressing larger volumes

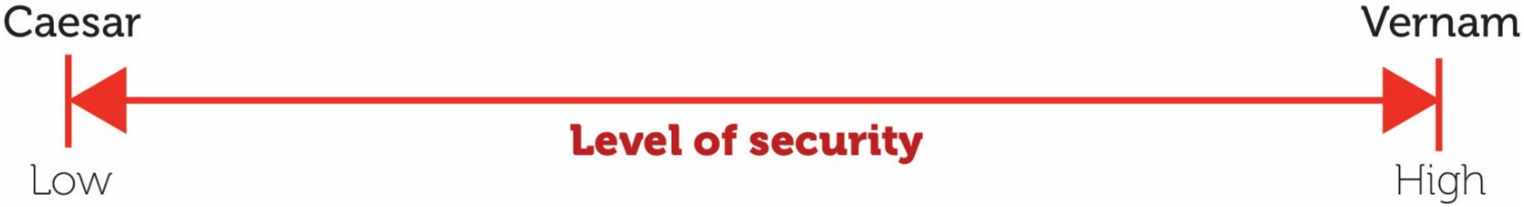
- In a text document each letter could be stored as an ASCII code of 8 bits
- In this document the word ‘*because*’ requires 56 bits of data (7 letters x 8 bits)
- Instead, the word could be added to a dictionary and assigned the binary code 01 which is a reduction of 38 bits for each occurrence
- A saving for 50 occurrences of the word:
  - 50 x 54 bits saving = 2,700 bits saved, or 338 bytes

Compression activities

- Complete **Tasks 1**, **2** and **3** of **Worksheet 1**

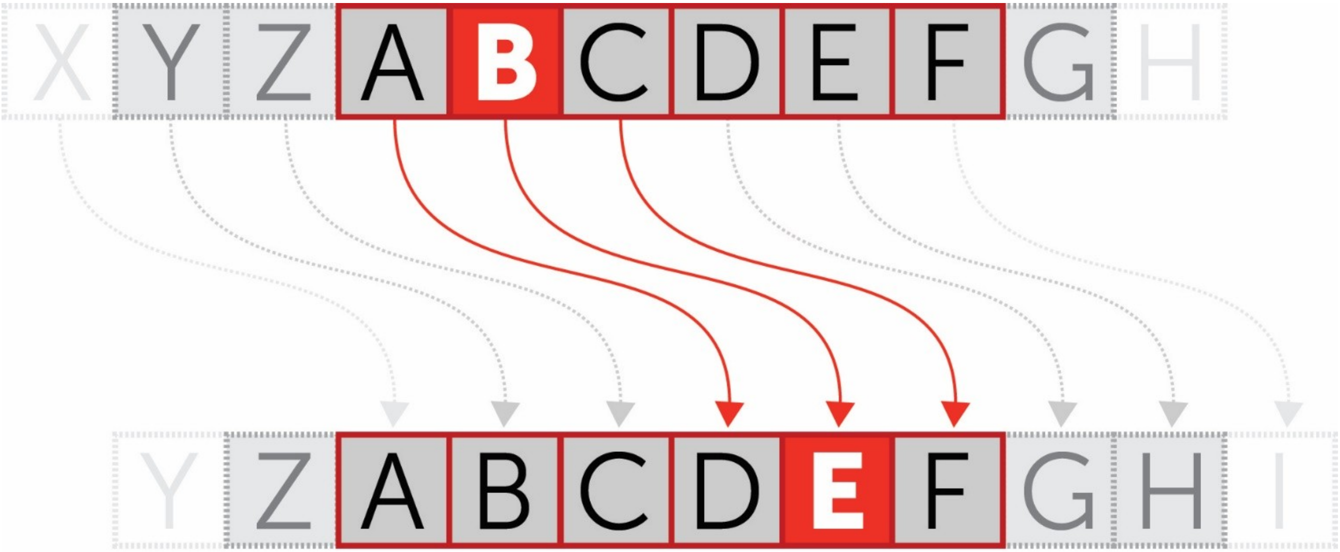
Encryption

- A way of making sure data cannot be understood if you don’t possess the means to decrypt it
  - Plaintext of a message sent is encrypted using a cipher algorithm and key into equivalent ciphertext
  - When received, the ciphertext is decrypted back to plaintext using the same or different key
  - Two methods at the opposite end of the security spectrum are the Caesar cipher and the Vernam



Caesar cipher

- The Caesar cipher is most basic type of encryption and the most insecure
- Letters of the alphabet are shifted by a consistent amount



Brute force attack

- A brute force attack attempts to apply every possible key to decrypt ciphertext until one works
- How many attempts might this take with the Caesar cipher?
  - Spaces are often removed to mask word lengths
  - Use the brute force method to decrypt the following: (Or, you could start by assuming vowels have the most occurrences)

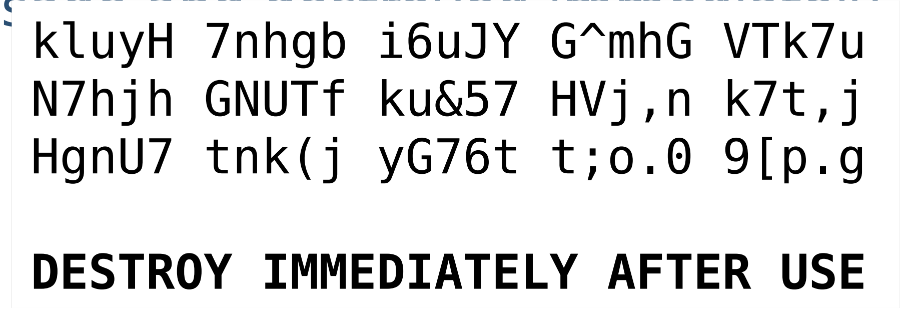
GAGDC NNQPV CTIGV

Frequency analysis

- Letters are not used equally often
- In English, **E** is by far the most common letter, followed by **T**, **A**, **O**, **I**, **N**, **S**, **R**, then **H**
- Other letters like **Z**, **J**, **K**, **Q**, **X** are fairly rare
- In Czech, the letter **Z** is only worth 4 points in Scrabble! It's worth 10 in the English version

Vernam cipher

- The encryption key, also known as the **one-time pad**, is the only cipher proven to be unbreakable
- The key must be:
  - a truly random sequence greater or equal in length than the plaintext and only ever used once
  - Shared with the recipient by hand, independently of the message and destroyed immediately after use

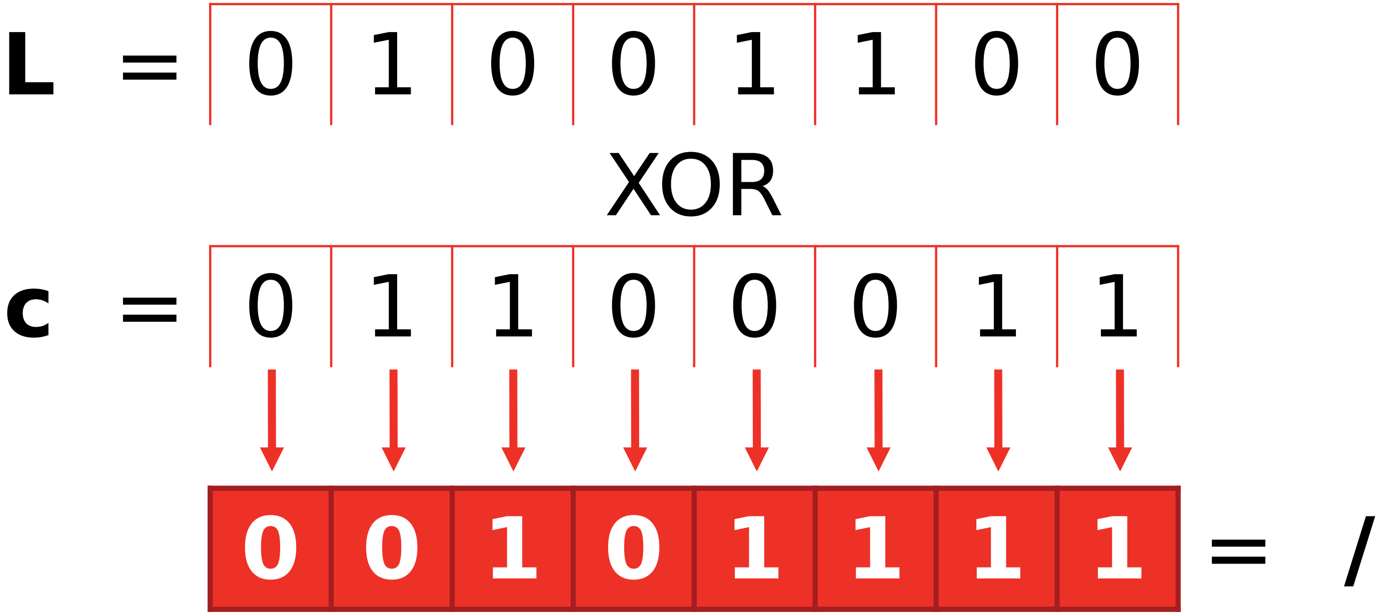


```
kluyH 7nhgb i6uJY G^mhG VTk7u
N7hjh GNUTf ku&57 HVj,n k7t,j
HgnU7 tnk(j yG76t t;o.0 9[p.g

DESTROY IMMEDIATELY AFTER USE
```

Decoding

- Encryption and decryption of the message is performed bit by bit using an exclusive or (XOR) operation with the shared key



L = 0 1 0 0 1 1 0 0

XOR

c = 0 1 1 0 0 0 1 1

0 0 1 0 1 1 1 1 = **/**

The one-time pad

- The one-time pad must be truly random, generated from a physical and unpredictable phenomenon
  - Sources may include: atmospheric noise, radioactive decay, the movements of a mouse or snapshots of a lava lamp
  - A truly random key will render any frequency analysis useless as it would have a uniform distribution
  - Computer generated ‘random’ sequences are not actually random

Activity

- Complete **Task 4** on **Worksheet 1**

Algorithmic security

- Ciphers are based on computational security
  - The keys are determined using a computer algorithm
  - A key derived from an algorithm, can also be unpicked
  - Given enough ciphertext, computer power and time, any key (except the one-time pad) can be determined and the message cracked

Symmetric encryption

- Symmetric encryption is also known as private key encryption
- The same key is used to encrypt and decrypt data
- This means that the key must also be transferred to the recipient
- What security problem does this pose?

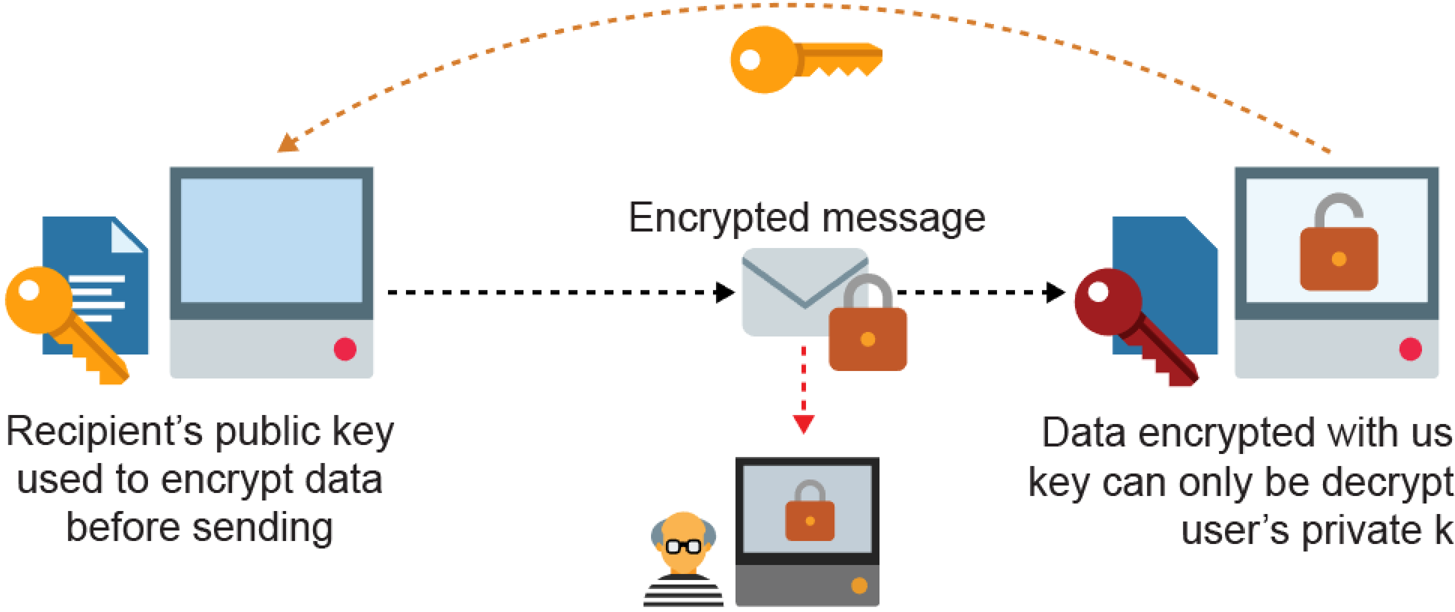
Symmetric encryption

- The key can be intercepted as easily as the ciphertext message
- This causes an obvious security problem
- For this reason, asymmetric encryption may be used instead

Asymmetric encryption

- This uses two separate but related keys
- One key, known as the public key, is made public so that others wishing to send you data can use this key to encrypt it
    - The public key cannot decrypt the data
    - A private key, known only to you, is used to decrypt the data

Asymmetric encryption



Recipient's public key made available to others wanting to send recipient data securely

Encrypted message

Recipient's public key used to encrypt data before sending

Data encrypted with user's public key can only be decrypted with the user's private key

Data can be intercepted but cannot be deciphered without the private key

The use of hashing

- A hashing function provides a mapping between an arbitrary length input and a usually fixed length or smaller output
- It is one-way; you cannot get back to the there original
- This is useful for storing encrypted PINs and passwords so that they cannot be read by a hacker
  - To verify a user's password, the software applies the hash function to the user input and compares the hashed result with the one stored

Plenary

- Encryption and compression change the contents of a source file for different reasons
- Lossy compression is most effective at reducing storage space
- Lossless compression maintains the integrity of the original data
- Encryption can be used to obscure a message

Copyright

Licence agreement